Quantitative Evaluation Methodology for Dynamic, Web-based Collaboration Tools

Chris Furmanski, David Payton, and Mike Daily
HRL Laboratories, LLC
(chris, payton, mjdaily}@hrl.com

Abstract

One of the primary hurdles to effectively evaluating adaptive software is that its very use alters the system from one moment to the next. Here, we developed and implemented an empirical-evaluation methodology that was successfully tested on an adaptive, web-based collaboration tool. Subjects participated in an "open-book" exam, spending limited amounts of time on the web researching a generic, non-technical topic, such as jazz or baseball, and were later tested on the information they had found. We employed objective, user-centered metrics to quantify subjects' ability to make enhanced web searches when using the tool. Test results from 8 subjects showed that our approach could reliably measure differences in a user's performance for different collaboration algorithms. Our protocol demonstrated that history-dependent tools can be evaluated without large subject pools and without extensive database preparation if careful evaluation design is utilized.

1. Introduction

Dynamic software and adaptive user interfaces offer a host of advantages over static systems by enhancements in system appearance, content, storage, usability, and/or functionality. However, the process of evaluating such adaptive applications is challenging because tool use during testing inherently alters the tool's performance.

Under the auspices of the DARPA Intelligent Collaboration & Visualization / Information Management (ICV/IM) program, HRL Laboratories developed an adaptive software tool to aid human collaborative activities [7]. During the initial development of PackHunter, ad-hoc tests of system functions were performed, although carefully controlled user evaluations were not conducted because of difficulties encountered trying to evaluate a dynamic system. The work described here developed a robust evaluation protocol for a unique class of adaptive collaboration-enhancing software and utilized this protocol to test a previously developed history-dependent collaboration tool.

PackHunter interface functionality

- Select a group to share trails with
- Highlight current user locations on paths
- Mark pages for others as interesting
- Jump to pages visited by others
- Make user paths overlap at common pages or view them independently

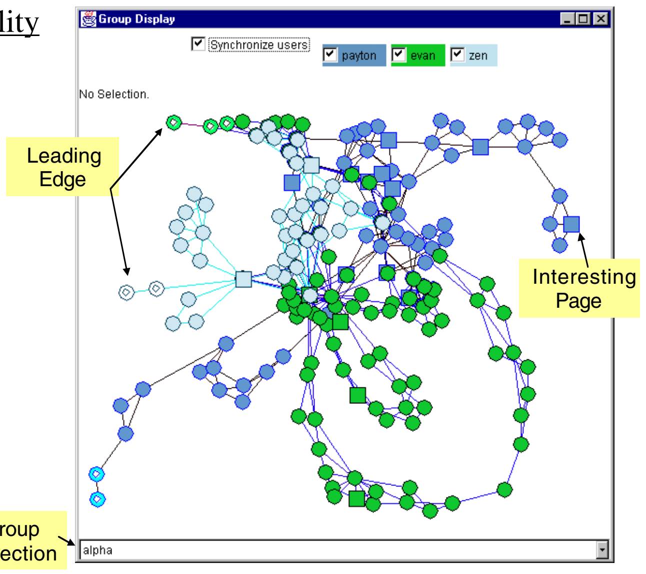

Figure 1: PackHunter's "collaborative browsing" interface used to visualize different websites (shapes) visited by different users (colors) in a collaborative environment.

1.1. History-dependence

We term the dynamic nature of adaptive tools "history dependence," acknowledging the fact that the very use of the tool alters the tool performance from one moment to the next. PackHunter, uses algorithms that are based entirely on techniques for mining information from the stored and current click streams of its users. As a consequence, the performance of PackHunter depends on both the *retrograde* activities of people who have used the tool previously and the *anterograde* nature of the tool that once it is used, the tool becomes altered for the next set of users.

Another important consequence of history dependent systems is that they are sensitive to both the quantity and quality of historical data, often showing a *critical-mass effect*, where meaningful outcomes are not observed without a sufficient amount of prior use that constitutes normal or representative use of the system.

Finally, we distinguish between two types of history dependent effects: first-order effects (the result of users' actions that directly alter a history dependent tool or its databases) and higher-order effects (those that arise in a collaborative environment when the course of a user's events are influenced by other users' actions). In the case of PackHunter, it is expected (and in fact, desired) that the tool should exhibit both first and higher-order effects, changing both the toll and the way users interact with each other.

2. Prior work

The term "history dependence" has been identified as a major issue of experimental testing of interactive dynamic systems such as simulators and adaptive interfaces **[6]**. Newell and Simon (1972) comment that "because of the strong history dependence of the phenomena under study, the focus on the individual, and the fact that much goes on within a single problem-solving encounter, experiments of the classical sort are only rarely useful. Instead, it becomes essential to get enough data about each individual subject to identify what information he has and how he is processing it" (p. 12). Prior use of empirical evaluation has been implemented in web-based navigation using adaptive hypertext based on dynamic user modeling where hypertext available to the user changes based on experience **[4]**. Practical guidelines, generic experimental design issues, and general methodological suggestions have also been outlined for user-adapted user-model systems [2]. Incremental testing procedures have been adopted as a model for object-oriented programming of adaptive software, where alternations in software class structures may change over time **[5]**.

3. PackHunter testbed

PackHunter uses the click streams collected in collaborative web browsing to find users with similar click streams, thus enabling the user to discover new potential collaborators and/or someone who might be working on a related problem. This evaluation focused on two of PackHunter's primary features geared towards at enhancing web-based collaboration: (1) collaborative browsing: a function to help users perform coordinated team exploration of hyperlinked data and (2) related collaborators: a function aimed at helping users find others who share their interests.

The collaborative browsing (CB) feature was designed for people who already know others with whom they need to collaborate. For these people, PackHunter provides the means for users to monitor the click streams of their group or team in real time. This real-time collaborative browsing feature allows users to join an interest group by interacting with browsing activity that represented visually by trails that consist of nodes and links that represent web pages and associations between web pages (as shown in Figure 1).

The find-related collaborators (RC) feature provides aspects that are complementary to collaborative browsing. RC enables users to identify others with common interests based on passive observation of their information access patterns. This can serve as an important tool to aid rapid formation of ad-hoc teams. The RC mechanism in PackHunter works through real-time analysis of trail histories collected from the CB tool.

4. Evaluation issues

The primary goal of this evaluation was to determine how PackHunter impacted a user's information gathering ability relative to not using the tool. A major challenge of the evaluation was to measure the degree to which the two independent experimental conditions (RC and CB) could be compared to a tool-less baseline /control condition. The selection of the proper metrics was critical for effectively measuring human behavioral performance (in the context of determining how PackHunter's features improved people's information-gathering performance). However, the operational definition of descriptors such as "improvement," here is an important prerequisite to defining the appropriate metrics.

4.1. User-centered judgments

Judging the amount or quality of useful information contained on any given web page is not trivial. Information quality is directly relevant because it could serve as a valuable metric for determining if PackHunter improved user's search performance. Techniques, using information theory for example, are used to quantify quality and quantity information in technologies such as search engines [3]. Algorithmic quantification of information is often an "information-centered" approach (i.e., focusing solely on the information being analyzed (such as on web pages) but is independent of the user's interpretation of the data).

Instead of using information-centered techniques, we employed a "user-centered" approach that uses human performance as a second-order estimate of information content. User-centered measures take the user's understanding of the information into account, which better reflects the desired outcome in the real world; measuring how much the tool improved *user knowledge* is the real goal, not simply measuring increased information content on a web page. Such user-centered techniques are commonly used in cognitive psychology, where particular measures of human behavior are used as proxies for psychological/cognitive processes that are not directly observable; the human user serves as a "black-box" in which behavior serves as an indirect measure of information quality. Our user-centered approach was particularly useful in evaluating PackHunter because it allowed for collection of information improvement (due to the tool) in measures that were directly relevant to web use.

User-centered methodologies can be classified as either subjective or objective measures. We considered several subjective metrics to measure user behavior, including measures of participant's qualitative measure of tool utility, report generation ratings (in which participants wrote reports based on information found and then rated the quality of other user's reports), and user ratings of the suitability of collaborator matches found by the tool. Ultimately we decided to utilize more objective measures that allow for improved experimental controls and more robust statistical treatments.

4.2. Objective measures

Since we opted to take a more objective approach to quantifying information quality, we considered a wide range of objective tests of user-centered information quality. Objective tests we considered included a series of fact-finding trials (a "treasure hunt" design) in which participants were instructed to find as many predefined facts (or a series of answers to very specific questions about a range of topics) somewhere on the web in a limited time (e.g., "What is the distance in KM between Cairo and Honolulu?" or "What is the total land area of Belarus?"). This approach was attractive in that it would give us a quantitative performance metric (i.e., how many questions the participant got right), however it seemed that specific, well phrased search engine queries would likely produce the answer to such trivia questions without lengthy searches thus eliminating the need for the tool. Other problems existed since this approach might cause user-search activity to focus on very narrow pockets of information, and search performance might have too heavily relied on the ease to which the specific questions could be answered more than anything specifically related to our tool.

A test-like measure had many attractive features (e.g., questions could be time limited had objective correct or incorrect answers, etc.), but concerns that the specificity of question topic had would significant impact on how well the tool would perform. For example, very specific question types, such as in a "treasure hunt" design, might not be well suited for measuring RC or CB features, because simple search engine queries could provide answers more quickly than PackHunter.

Other question types, such as questions with ambiguous words that might confuse search engines (e.g., the query "When did Bill Clinton last visit New Jersey?" is ambiguous to a search engine as many of the items returned will have to do with legislation (bills) in the city of Clinton, New Jersey) might play more into PackHunter's strengths, but these types of questions were tough to operationally define and it was difficult to generate a number of these questions without first determining if each question does cause search engines to fail (i.e., did the search engine fail to produce the correct answer in the first few pages of hits). Another fear was that because the RC function required significant overlap between users searches (i.e., visiting the same websites or searching for the same type of information), it would be difficult to generate a series of such questions with the additional constraint of having to have some of the question topics be related.

In the end, we decided that searches guided by open-end topical research questions (e.g., "Find out everything you can about jazz,") would provide a more thorough type of search that could not be entirely bypassed by simple keyword search-engine searches. Further, instead of providing participants with a list of questions (that would be used as the objective measure) before a search session, in this case, participants were instructed to find as much information as they could about a the general search topic (e.g., jazz or baseball).

Our approach was paired with an instruction phase in which participants were informed that a test would follow in which they could only use web pages found during their search to answer the test questions. The specific test questions were only provided *after* the participant completed the search. Thus, by providing the questions after the search, participants could not simply find the needed answers with a search engine. Performance on the test would reflect how much information participants were able to find using the tool. To assure that the test evaluated the tool and not the user's memory capacity, we allowed the user to refer back to any web pages encountered during their search. This approach was analogous to an "open-book" exam, in which participants used existing information (i.e., web pages previously found during the general topical search) to answer the questions but were prevented from looking for new information.

4.3. Background database

The development of a practical evaluation methodology involved overcoming the complications involved with history dependence. PackHunter must have material in its database in order to produce meaningful results. Thus, one of our primary challenges was to develop an evaluation methodology that, while constrained by a limited time schedule and limited user population, avoided the need for a very large database containing the search histories of thousands of users.

This critical-mass effect is particularly a problem in relation to the use of PackHunter's find related collaborator (RC) function. The RC algorithm is computed entirely from data gathered by prior users, so in order for this feature to work, a sufficiently large database of prior users is required. The primary mechanism used to find new information with RC is "scent diffusion." In order to find related information between users, scent diffusion requires multiple users to have visited some of the same web sites (thus spreading the user's "scent" to other pages). Therefore, in order for RC to be effective, there must be some overlap in several users' search history. With just a few users, and just a short time to develop extant browsing histories, it was clear that open-ended searches on the web were very unlikely to create any significant overlaps of user trails.

We considered several options for developing a usable background database. One idea was to recruit an existing large user community, such as newsgroup users or medical researchers, to use PackHunter. Another idea was to link the software to an internal proxy server, so that PackHunter could, in the background, track all web usage for HRL employee volunteers. However, such an approach would not incorporate the ability of users to rank pages (another type of PackHunter functionality) nor would it allow users to define the topics of particular searches. Had there been more time, many of the problems could have been solved and these approaches might have been more feasible.

We finally decided that the best way to address the critical mass problem for our evaluation was to limit search to particular subject domains. We identified a set of subject domains (e.g., French Impressionism or Jazz) such that each domain established a unique subset of information that was narrow enough to limit the breadth of information a user might encounter, but broad enough to allow for a wide range of distinct search activities. Within each subject domains we selected a number of specific topics that might or might not overlap depending on semantic relations between these topics.

To generate our background database, we developed a strategy whereby a number of users who were not part of our test subject pool were each given a topic to investigate within one of our selected subject domains. For example, given the subject domain of French Impressionism, users were asked to gather all the information they could find on topics related to French Impressionism such as: Monet, light, Fauvism, galleries, Paris, etc. While each of these topics alone could be quite general, focusing the search within the confines of the particular subject domain of French Impressionism greatly focused the types of web pages that would be encountered. It also ensured that some degree of overlap would occur between the search trails through these topics. We made sure that none of the users who participated in generating the background data were used in the final evaluation. This way, no prior experience gained with respect to a particular background topic would be likely to influence a subject's performance.

Normally, PackHunter's CB function is used by a group of people looking simultaneously for information about the same topic. However, our open-book test-taking format needed to disregard the multi-user interaction aspects of CB because allowing users to view the search activities of prior users could have impacted the information gathering performance of a new user. To prevent this problem, the background data for tests of the CB function were generated by user's searches on the specific topics that would eventually be given to the test subject. At testing time, these search results were presented to actual test users via the CB function so as to appear that numerous other collaborators were already working on the same search task.

4.4. Freezing the background

It was also necessary to establish a way to nullify history-dependent changes that occur to the database when it was used from one test user to the next. We also needed to ensure that the background database would not change from one user's trial to the next; we did not want the activities of a user in one trial to indirectly alter a later user's trial. Similarly, during the evaluation, we wanted to be able to have different users perform searches on the same topic without being able to see that some other user had done the exact same search sometime before them. To obtain these capabilities, we added save and load features to PackHunter. These features allowed us to extract all of the relevant information in the PackHunter database and save it to an output file. At any time later, we could reload this information back into the database and restore the tool to its previous state. Incorporating such a feature in the development of future adaptive / history-dependent systems would be strongly advised. For the purposes of our experiments, we saved versions of the database that only contained the background searches. When beginning new evaluation trials we simply reloaded these saved databases to serve as the same, controlled starting point for each user.

5. Empirical evaluation

We decided on a standard experimental design in which user performance for the test conditions (in this case RC and CB) were compared to control conditions (where participants performed searches using the same PackHunter interface but not using the RC or CB functionality). Thus, the basic contrast compared participants' performance on the "open-book exam" in experimental conditions (after using RC and CB) with participants' performance on a similar exam in which participants used the tool (but not the added RC/CB functionality) for the control conditions. If participants answered more questions correctly using either RC or CB than in the control conditions, we would conclude that the experimental conditions provided users with better information when using the RC or CB functionality than when they did not.

We decided upon a within-subjects design in which each participant perform all conditions (as opposed to a between-subjects design, which has distinct participants for each condition). While a within-subjects design would take more of each participant's time, it had several key advantages. First, because each participant performs each condition, it drastically reduces the total number of participants required (in this case by a factor of 4). This design also has the added bonus of reducing the between-subject variability (the natural variance that arises from between different individuals), giving certain types of statistical comparisons more statistical power.

5.1. Testing approach

Our basic experimental design included 2 experimental conditions (RC and CB) and 2 control conditions. Thus for each condition, a separate search topic would be required (i.e., 4 distinct topics would be needed). Keeping in mind that PackHunter is a web-based tool and that our testing procedure used an "open-book" exam as the primary measure of participant's performance, we were forced to find search domains that had both an extensive exposure on the web and numerous related topics with which we could populate the database.

We first settled on four broad topical groups/categories (French Impressionism, Jazz, Baseball, and Silent Film) from which specific topics would be chosen. Each group was assumed to have roughly the same information content and internet exposure (no norms for internet-based information content could be found). Within each group, a single search topic was chosen and randomly assigned to a particular condition pairing (either RC-Control or CB-Control). Each of the four search topics were confined to famous people that were representative of each group/category selected. We intentionally restricted the topics to famous people so that similar question syntax could be used for each topic despite the specifics (e.g., when was person X born?). The topic pairings selected (randomly) were Charlie Chaplin (silent film star & director) and Babe Ruth (baseball player) for the CB-control condition, and Edouard Manet (French Impressionist) and Louis Armstrong (jazz trumpeter) for RC-control.

5.2. Question format

After selecting the specific topics (individual famous people), lists of questions were generated. Each list consisted of 30 questions and was based on either biographical (e.g., "When was person X born?") or career-oriented information (e.g., "When was Y's professional debut?"). While questions were intended to be approximately of equal difficulty, differences in difficulty between question lists were negated using standard counterbalancing techniques (described below).

The questions were generated as multiple-choice questions; participants did not have to generate the correct answer, but instead, they were to choose from a pair of answers, one of which was correct. This multiple-choice approach had several advantages. First, the goal of this testing procedure was to quantitatively judge the quality of the information participants found using the tool, not to measure the test-taking ability of the participants, per se. Thus, we wanted to make the testing procedure as simple and as straight forward as possible.

Second, because participants performed four topical searches (and thus, had four tests to take) we hoped to limit the amount of time participants would need to dedicate to the actual test-taking procedure. So, by providing participants with possible answers, we hoped that the confined search space during the testing procedure would speed up their search, thus limiting the evaluation procedure to a reasonable amount of time. It should be noted that participants were not given the questions (or answers) before the search procedure and were instructed not to guess; this will be addressed in more depth, below (see Testing Procedure).

We considered a large range of alternative testing formats, including simple "fill-in-the-blank" questions and a more subjective approach that used peer-reviewed report writing. We also thought a good approach would be to generate an algorithm that would automatically search the text from the web pages found by participants. This approach would have alleviated the participants from actually having to perform the tests, thus avoiding variability between participant test-taking strategies. This algorithm approach would have also been more efficient as it would have taken less time per participant, but this automated system was never implemented due to lack of time for algorithm development.

5.3. Condition counterbalancing

For a given participant, topics were assigned to particular conditions following standard counterbalancing methods (in this case, it was a modified Latin-square design) [9]. The goals of counterbalancing are twofold. First, counterbalancing assures that the conditions are presented in different orders (in a controlled fashion) so to avoid confounding results that might arise from running in a particular order (so called order effects). Second, counterbalancing is used to assure that each topic is equally assigned to the various conditions, so the final results (calculated by determining differences between the experimental and control conditions) are not confounded by differences in the difficulty of the topics assigned to each condition. The following table displays how each topic was paired with each participant in a counterbalanced manner across 4 different participants.

It should be noted that a complete counterbalancing (where each topic is assigned to each condition) could not be achieved here

Orders	1	2
1	VR-EM	Control-EM
2	Control-LA	VR-LA
3	CB-CC	Control-CC
4	Control-BR	CB-BR
	3	4
1	CB-BR	Control-BR
2	Control-CC	CB-CC
3	VR-LA	Control-LA
4	Control-EM	VR-EM

Table 1. A modified Latin-square counterbalancing for different orders (columns) using combinations of different conditions and different search topics used between subjects (rows).

because only two background databases could be generated in the allotted time (instead, three would have had to be generated - see Specific Database Generation). However, because the critical comparisons were between each of the experimental conditions (RC and CB) and the control groups (in this case, one control group was assigned to each experimental group), suitable counterbalancing was achieved within a pair of conditions (RC-control or CB-control).

5.4. Specific database generation

Because PackHunter is a "history dependent" tool, we needed a functional database with prior searches in order to use the key RC and CB features of the tool. To create suitable background data for testing the RC function, we chose to populate two of our pre-selected subject domains: Jazz and French Impressionism. We had a group of volunteers perform a series of ten independent directed searches on a range of specified topics within each of these categories. Care was taken in the choice of topics to assure that they were not identical to the topics that were to be used in testing. Since the volunteers were not expected to make use of PackHunter RC or CB features, several people could perform these searches in parallel without influencing each other. This allowed for a rapid and homogenous completion to populating the database.

Creating the background data for the CB function was a simpler task. For CB, we needed several searches that were from both the same subject domain and on the exact same topic. We selected two such topics for generating the CB background data and did ten searches for each. We used Charlie Chaplin (CC) from the Silent Film universe and Babe Ruth (BR) from the Baseball domain as our two topics.

5.5. Outcome and limitations

Our attempts to fill the background database were sufficient to obtain independently counterbalanced experiments for CB and RC. Such a design is adequate to independently test CB and RC against control conditions, but is not suitable for comparing CB and RC with each other. In this design, two sets of background searches (on two different domains) were collected for RC tests, and two sets of topic searches were collected for CB tests. This gave us two topics to test with RC and two to test with CB. The independence of these tests meant that they each had to have their own control conditions. Because of this, it was necessary to test each user under two control conditions in order to obtain complete counterbalancing.

Had we been able to populate our background database more fully, each topic would have been used both as a control condition and an experimental condition (one typical type of counterbalancing). This way we would have only needed 1 control condition (instead of 2) which would have required less testing for each participant. Under that more ideal design, each of the two experimental conditions would have been compared to the same control condition, requiring data to be generated for 3 topics (one for the control and two for the experimental conditions). Each of the databases would have been assigned to different conditions (RC/CB/Control) in a between-subjects manner in order to account for potential difference in the difficulty/quality of each dataset.

5.6. Testing procedure

Participants were 8 graduate students (5 male, 3 female) that participated for monetary compensation. All participants signed voluntary consent forms.

5.6.1. Introduction phase

The experiment began by introducing the participant to the PackHunter interface. The main interface (which displayed a spatio-graphical representation of the search history) remained open and was used in conjunction with a web browser (Netscape Navigator 4.x) that was also open during all phases of the evaluation. Participants were shown how to interact with the interface, what the symbols on the interface meant, how to mark useful pages, and how to use the other PackHunter features (i.e., CB and RC). Participants were also given 10 minutes to perform a supervised sample search (on the topic of NASA's Skylab project) in which the participants could get hands-on familiarity with the interface and its features. These data were not included in the data analyses nor saved to a database (as to not affect subsequent searching).

5.6.2. Searching phase

After the Introduction phase, participants were given a set of instructions to read while the experimenter verbally explained the same instructions. The instructions explained that the participants were to use specific features of the PackHunter tool during certain times during one of four successive web searches on particular famous people. Participants were told to find as much material on each person as possible as an "open-book" multiple-choice test would follow. The participants performed four successive searches on the topics assigned in a particular order (determined from the counterbalancing) before starting the testing phase.

During the search, participants were also instructed to annotate particular web pages as useful or important using additional PackHunter page-marking capabilities. This allowed the participants to return to these pages during the test phase, enabling the participants to answer the subsequent test questions. Participants were also informed to work quickly and that they didn't need to learn or memorize any of the information because it would be accessible to them during testing. Participants were given up to 60 minutes to perform each of the four searches (although no participant used the entire time).

Participants performed 1 search using the CB feature, 1 search using the RC feature, and 2 controls searches which used the tool interface for tracking and page marking, but did not use either RC or CB features. Before each search, the generic baseline database was reloaded to assure that the history-dependent nature of PackHunter utilized the same collection of information at the start of each search. The baseline database contained all of the background searches performed beforehand by volunteers (see Database Generation) that allowed the CB and RC features to work. After each search, the new information was saved out to a separate database, and the old baseline database was reloaded before the next search.

5.6.3. Testing phase

After the participants finished all four of the searches, participants began the testing phase of the evaluation. The testing phase either occurred on the same day or within 4 days of the searching phase. At the beginning of each test, the appropriate database, which contained the participant's prior searches, was reloaded. This allowed the participant to view the spatio-graphical representation of their prior web search, including the pages they marked as important. Participants could then simply click on a desired symbol in the spatio-graphical representation and the corresponding webpage would appear in a browser opened in another window.

The participants read the instructions for the testing phase and were also given a verbal explanation by the experimenter. During the testing phase, participants were instructed to work as quickly as possible, but above all else, they were instructed not to guess on any question. They were instructed to only answer questions to which an answer could be found on one of the previous websites they had visited. In order to assure that participants did not guess on a question, the participants were also instructed to write the URL of the page they used to find the answer to the question. (This URL could be used to verify their answers and also provided some idea as to how many of the websites were actually useful for this sort of task.) Other approaches for this verification process could have incorporated a web-based form into which participants could have cut and paste URLs (which might have saved the participants time during testing).

Participants were also given examples of test strategies and, for the sake of uniformity, were instructed to select a single strategy, using the same strategy for each test. One example strategy was to use the browser's find feature to search for key words (from either the question or either of the answers) in a given web page to reduce the amount of time participants might need to visually scan a webpage to see if it contained the relevant information. Participants were given 20 minutes to take each test, and were also required to count the number of questions answered in 5-minute intervals. This would provide an estimate of whether the time limit of 20 minutes was sufficient to exhaustively explore the participant's collection of web sites found during the search phase. Such data could distinguish if the participants had exhausted their ability to find pertinent facts (i.e., they might not be able to answer any more questions between 15 and 20 minutes) or if they were still finding relevant information (i.e., they continue to answer more questions after each interval).

5.7. Results

The main focus of this evaluation was to test if, all other things being equal, particular features of PackHunter, namely the View-Related (RC) and Collaborative-Browse (CB) features, improved the quality and content of specific information found through searches on the web. In our evaluation, information quality was objectively quantified by measuring participant's performance on a series of multiple-choice tests based on their web searches. Thus, after the 8 participants finished the testing procedure, both the test performance and web-usage statistics (e.g., number of pages visited, marked as important, time spent per page, etc.) were analyzed to see if differences in participant's test performance existed between the experimental (RC & CB) and control groups.

The key finding is presented in Figure 2. We found the (RC) searches yielded a reliably higher mean performance (64.6% (19.375/30 questions)) than did the control group (52.5% (15.75/30 questions)), t(7)=3.34, p=0.01238 (paired, 2-tailed t-test with 7 degrees of freedom).

Analysis of the collaborative-browsing (CB) searches (50.4% (15.13/30)) trended towards being worse than the corresponding control

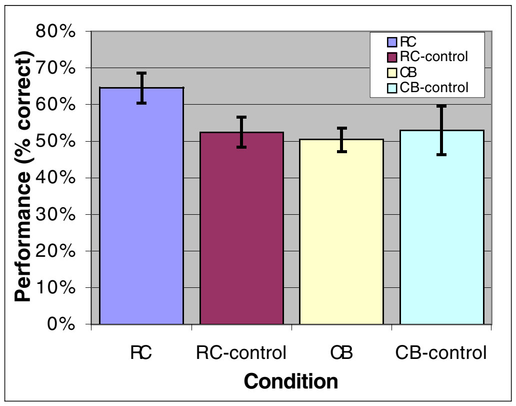

Figure 2: Mean performance for all conditions.

group (52.9% (15.875/30)), but the differences were not statistically different, t(7)=0.30, p>0.05.

Analysis of performance for given topics showed that there was not a reliable difference in difficulty of the tests used in RC-control conditions: between EM (16.75/30) and LA (18.5/30), t(7)=1.18, p>0.05. Analysis of the other groups (CB & control2) showed that BR (18.25/30) was reliably easier than the CC topic (12.875/30) (t(7)=3.80, p=0.0067). However, because the topics were counterbalanced across the CB and CB-control conditions, this did not impact the results of the tool evaluation.

6. Discussion

The two major goals of this evaluation were: (1) to generate an evaluation methodology for a malleable, history-dependent piece of software, and (2) to test how a specific piece of web-based software (i.e., PackHunter) performed. Our approach accomplished both tasks. This paper outlined one route for evaluation of a history-dependent application that included small sample sizes, within-subjects experimental designs, objective user-centered metrics, and tool-manipulations that allowed the database to be reset to overcome problems with adaptation involved with iterative testing. We also demonstrated that a specific feature of a specific tool (i.e., the find related collaborator feature of PackHunter) reliably improved participants' performance on test questions when compared to the control test that did not use the RC feature.

Participants' improvement on test performance when using RC suggests that participants were able to find more relevant information when they had access to related but not identical searches in the same general topic area. These results support the notion that dynamic, history-dependent tools can be evaluated without using a large subject pool if careful planning is made in the development of the experimental protocol and if carefully structured, multiple background databases can be used.

We were surprised the CB function, that provided users with information about other participants' prior searches on an identical topic, did not improvement searching performance. Yet, for those users with the RC function, that had the benefit of seeing the searches of other users in a related but not identical topic, did show improvement. It is possible that seeing websites and information others have previously explored with the CB function made users less likely to explore on their own; maybe there was a creative aspect to seeing results from searches of similar but not identical topics that stimulated people to expand their exploration of alternatives beyond their normal horizons.

Future work could focus on obtaining a better understanding of the effects of closing the loop in a history-dependent tool such PackHunter. Since one of the ultimate goals of the PackHunter tool was to create a positive feedback cycle, whereby information provided by the tool at one point in time impacts the information it can provide in the future. Positive feedback cycles such as this occur in nature in the form of autocatalytic systems and can even be observed in economic systems [1]. If we are to truly understand the impact of a tool such as PackHunter, we may need to understand the dynamics of naturalistic positive feedback cycles as well. In fact, our current methodology (involving saving and reloading of fixed databases) precluded an evaluation of the positive-feedback effect. Further research is recommended to develop such a methodology and to see if these positive feedback effects do in fact amplify the gains provided by PackHunter.

Future work should also include a blend of information-centered metrics that could be used in conjunction with user-centered metrics. While information-centered metrics may be useful during future evaluations of PackHunter and other history dependent tools (since it provides an unbiased estimation of information content), the user-centered metrics used here gave us the best compromise of efficiency and ease-of-implementation. Other metrics focused on measuring tool performance across other dimensions, such as subject-matter (topic) specificity or the amount of necessary user-user overlap required to produce useful information are also an empirical questions that may provide for fruitful future research.

In the end, our evaluation protocol overcame many of the difficulties that arise when performing empirical evaluations of malleable software, such as first- and higher-order history dependence, retrograde and anterograde effects, and critical-mass effects. The development of our evaluation procedure was also pragmatic, as it took into account realistic constraints of rapid, real-world software development cycles, such as brief deadlines and small test-user populations. Ideally, our empirical evaluation would have utilized more subjects, more test conditions, and been sensitive to positive-feedback effects, but nevertheless, we attained our expressed goal of defining and implementing a feasible set of methodological procedures and metrics designed to circumvent the complexity of evaluating adaptive, history-dependent tools.

7. References

[1] Arthur, B. (2000) Positive feedbacks in the economy. Scientific American, 262, pp. 92-99.

[2] Chin, D. (2001) Empirical evaluation of user models and user-adapted systems. *User Modeling and User-Adapted Interaction,* 11, pp. 181-194.

[3] Henry, P. (1998) User-centered information design for improved software usability. Artech House.

[4] Kaplan, C., Fenwick, J., and Chen, J. (1993) Adaptive hypertext navigation based on user goals and context. *User Modeling and User-Adapted Interaction,* 3(3), pp. 193-220.

[5] Keszenheimer, L. and Lieberherr, K. (1994) Incremental testing of adaptive software. Technical Report NU-CCS-94-22, Northeastern University, November, 1994.

[6] Newell A. & Simon H. (1972) *Human Problem Solving*, Prentice-Hall.

[7] Payton, D., Daily, M., and Martin, K. (1999). Dynamic Collaborator Discovery in Information Intensive Environments. *ACM Computing Surveys,* 31(2), June 1999

[8] Höök, K., Karlgren, J., Waern, A., Dahlbäck, N., Jansson, C., Karlgren, K., Lemaire, B. (1996) A glassbox approach to hypermedia. *Journal of User Modeling and User-Adaptive Interaction*, 6, pp. 157-184.

[9] Woodward, J., Bonett, D., and Brecht, M. (1990). Introduction to linear models and experimental design. Harcourt Brace Academic Press, Florida.